

# TABLE OF CONTENTS

ZZZZZ	OOOOO	BBBB	EEEE	X	X
Z	O O	B B	E	X	X
Z	O O	B B	E	X	X
Z	O O	BBBB	EEE	X	X
Z	O O	B B	E	X	X
Z	O O	B B	E	X	X
ZZZZZ	OOOOO	BBBB	EEEE	X	X

ZZZZZ	888	000
Z	8 8	0 0
Z	8 8	0 0
Z	8	0 0
Z	8 8	0 0
Z	8 8	0 0
ZZZZZ	888	000

CCC	PPPP	U	U
C C	P P	U	U
C	P P	U	U
C	PPPP	U	U
C	P	U	U
C C	P	U	U
CCC	P	UUU	

ZOBEX  
P. O. Box 1847  
San Diego, California 92112  
(714) 571-6971

## TABLE OF CONTENTS

WARRANTY.....	3
CHARACTERISTICS AND FEATURES.....	4
BOARD SET UP.....	6
PORT ASSIGNMENTS.....	8
PROGRAMMING.....	9
THEORY OF OPERATION.....	16
BUS SIGNALS USED.....	16
THE ZOBEX MONITOR ROM.....	23
TROUBLE SHOOTING TIPS.....	27
APPENDICES.....	28

## WARRANTY

ZOBEX warrants its Z80 CPU boards to be free from defects in materials and workmanship for a period of six months from purchase date. ZOBEX will, at its option, repair or replace the defective part or parts to restore the board to proper operating condition. All such repairs and replacements will be made without charge for parts or labor when the board is returned (postage paid) to ZOBEX.

This warranty does not cover boards which have failed, in the judgement of ZOBEX, as a result of alteration, accident, abuse, negligence, improper supply voltages, or misapplication.

This warranty is in lieu of all other warranties, expressed or implied, including warranties of merchantability or fitness for use. In no event will ZOBEX be liable for incidental or consequential damages arising from or in any way connected with the use of this product.

## The ZOBEX Z80 CPU

The ZOBEX CPU is an IEEE standard S-100 module that incorporates processor, EPROM, lots of I/O, real-time clock, and vectored interrupt controller, all on one board. This powerful and flexible board is unmatched in quality and features, and is adaptable to a wide variety of home and OEM applications.

### CHARACTERISTICS AND FEATURES

Z-80A CPU, with clock rates of 2 and 4 MHz, switch selected. The clock speed switch may be changed at any time, even if the computer is running.

Compatible with the IEEE S-100 bus standard, and also with pre-standard computers, since PWAIT, and Z-80 memory request and refresh signals are provided.

Switch-selectable power-on jump, to any 4K boundary.

Software-readable 8-pole switch (Info-switch), for setting options such as baud rate, terminal type, etc.

Vectored interrupt controller on-board, handles all 8080 and Z80 modes. Complete software control over interrupts, including individual masking, priority selection, and polarity.

M1 wait state to accommodate slow memory.

On-board EPROM, which can be made to disappear under program control, thus making the entire 65K of address space available when desired.

High quality I/O connectors that won't vibrate loose or fall out.

Four on-board serial ports, using ZILOG DART UARTs, featuring total software control of modes, modem signals and baud rates to 38,400 baud. If synchronous communications are required, one or more of the DARTs may be replaced with ZILOG SIO chips.

External clock may be used to 800,000 baud.

Three 8-bit parallel ports, using the INTEL 8255. Configured for CENTRONIX parallel interface. This board will drive the EPSON printer directly.

Real time clock, using the ZILOG CTC, interrupts the CPU at software-selectable rates. Interrupts from the real time clock and the serial ports are software selectable. This board will run modern interrupt driven operating systems such as MP/M, with no external circuitry.

Bus status signals are placed on the data bus during pSYNC time for compatibility with older boards.

The MWRITE signal may be generated either on-board or externally.

Configuration options are selectable by jumper for wait states, on-board or external I/O ports, and EPROM disable.

EPROM monitor includes a disk bootstrap for the ZOBEX disk controller and many debugging tools and utilities.

Six months warranty on parts and labor.

With so many features on one board, the ZOBEX CPU economizes considerably on power and heat, since there will be perhaps only three boards in the computer. Thus you can use a smaller box, with a smaller supply, and no noisy fan.

## BOARD SETUP

As normally shipped, the board is configured as follows:

Power-on jump to address F000.

EPROM enabled, located at address F000.

4 MHz clock rate.

No M1 wait states.

Serial I/O channel A (J3) set for 9600 baud (via monitor software).

Info-switch (IC50) FFFF (monitor doesn't use it)

This configuration is correct for the software supplied on the distribution diskette. If changes are necessary to suit your applications, they may be easily made as follows:

### POWER-ON JUMP

The power-on jump address is controlled by the dipswitch at IC36. The top four sections of this switch correspond to the top four bits (in reverse order) of the address of the 4K boundary desired. To make a bit ZERO, its switch must be ON (closed). To make a bit ONE, its switch must be OFF (open).

#### Power-on Jump Switch (location 36)

Jump Address	bit 12 SW1	bit 13 SW2	bit 14 SW3	bit 15 SW4	
0000	on	on	on	on	
1000	off	on	on	on	
2000	on	off	on	on	
3000	off	off	on	on	
4000	on	on	off	on	
5000	off	on	off	on	
6000	on	off	off	on	
7000	off	off	off	on	
8000	on	on	on	off	
9000	off	on	on	off	
A000	on	off	on	off	
B000	off	off	on	off	
C000	on	on	off	off	
D000	off	on	off	off	
E000	on	off	off	off	
F000	off	off	off	off	(ZOBEX monitor address)

## CLOCK SPEED

Section 5 of the same dipswitch (IC36) controls the CPU clock rate. For 4 MHz, IC36 SW 5 should be OFF. For 2 MHz, it should be ON.

## EPROM DISABLE

To disable the on-board EPROM, either pull pin 4 of IC 31 (74LS20) out of its socket, or cut the trace between the pads located to the right of IC 31 pin 4 on the back of the board.

## ON-BOARD I/O

To disable all the on-board I/O devices, pull IC 32 (74LS138) out of its socket.

## M1 WAIT STATE

To add an M1 wait state on all instruction read-up cycles, add a jumper between the pads located just left of pin 15 of IC 40 (9519) on the top side of the board.

## I/O PORTS

The on-board serial and parallel I/O devices, the baud rate selection, the clock interrupts, the interrupt controller, the Info-switch, and the EPROM disable features are all accessed from the software by means of I/O ports.

# PORT ASSIGNMENTS

FUNCTION	PORT	CONNECTOR
Serial Channel A		
Data in/out	00	J3
Status/control	01	
Serial Channel B		
Data in/out	02	J4
Status/control	03	
Serial Channel C		
Data in/out	10	J1
Status/control	11	
Serial Channel D		
Data In/out	12	J2
Status/control	13	
Parallel Channel A		
Data OUT	08	J5
Parallel Channel B		
Data OUT	09	J5
Parallel Channel C		
Data IN	0A	J5
Parallel Control A-C	0B	
Timer Channel 0	04 (Baud rate for serial channel A)	
Timer Channel 1	05 (Baud rate for serial channel B)	
Timer Channel 2	06 (Baud rate for serial channels C & D)	
Timer Channel 3	07 (Periodic clock interrupts)	
Info-switch		
Data IN	0C	
EPROM Disable		
OUT	0C	
Interrupt Controller		
Data in/out	0E	
Control	0F	

## PROGRAMMING

The ZOBEX CPU provides an extremely capable and flexible environment for system software through use of programmable LSI devices. The permutations and combinations of all the features offered by these powerful chips are nearly limitless. The manufacturer's data sheets for the DART, CTC, 9519 interrupt controller, and 8255 parallel I/O should be examined carefully to take full advantage of features.

Some typical programming requirements will be shown by example to introduce the concepts. The software contained on the ZOBEX system diskette should also be studied as examples of programming for the LSI devices.

### INITIALIZING THE SERIAL PORTS

The ZOBEX monitor initializes serial port A for typical CRT terminal mode upon booting up. However, if this mode is not what you need, your own software can re-define the mode, and set up the other serial ports (which the monitor does nothing with).

Initialize Serial Channel A for 9600 baud.

```
ld      a,18h          ;chip reset command
out     (1),a
ld      a,14h          ;select reg 4, reset interrupts
out     (1),a
ld      a,84h          ;select 32x clock, 1 stop bit
out     (1),a
ld      a,3            ;select reg 3
out     (1),a
ld      a,0clh         ;8 bits, enable rcvr
out     (1),a
ld      a,5            ;select reg 5
out     (1),a
ld      a,0eah         ;select DTR, 8bits, RTS, tx enable
out     (1),a
```

Alternatively, the Z80 otir instruction may efficiently be used to output a table of bytes to do the same thing, as follows:

```
ld      b,7            ;number of bytes to send
ld      c,1            ;i/o channel to be used
ld      hl,tbl         ;point to table to be output
otir                     ;send table until b = 0
-
-
tbl:    00011000b       ;reset interrupts
        00010100b,10000100b ;reg 4, 32x, 1 stop bit
        00000011b,11000001b ;reg 3, 8 bits, enable rcvr
        00000101b,11101010b ;reg 5, DTR, 8 bits, RTS, tx enable
```

## BAUD RATE SELECTION

The CTC chip must be programmed to generate the proper clock rates to the DARTs.

Table of values for baud rates (assumes 32x clock)

Baud	Value
38,400	1
19,200	2
9,600	4
4,800	8
1,200	32
300	128

The following code will configure channel 0 of the CTC to send the correct clock rate for 9600 baud to serial channel A.

```
ld      a,45h          ;no ints, counter mode
out     (4),a
ld      a,4             ;divide by 4 = 9600 baud
out     (4),a
```

## SERIAL IN and OUT

The following code is an example of a simple method to read and write bytes to the serial device connected to port A (usually a CRT terminal). The mode must have been previously set up correctly, of course, such as in the examples above.

Output a character from C register to Serial Channel A

```
outch:  in      a,(1)          ;get status byte
        and     04             ;mask out tx buffer bit
        jr      z,outch        ;loop back if not high (1)
        ld      a,c            ;get character
        out     (0),a          ;send character
        ret                     ;return from subroutine
```

Input a character to A register from Serial Channel A

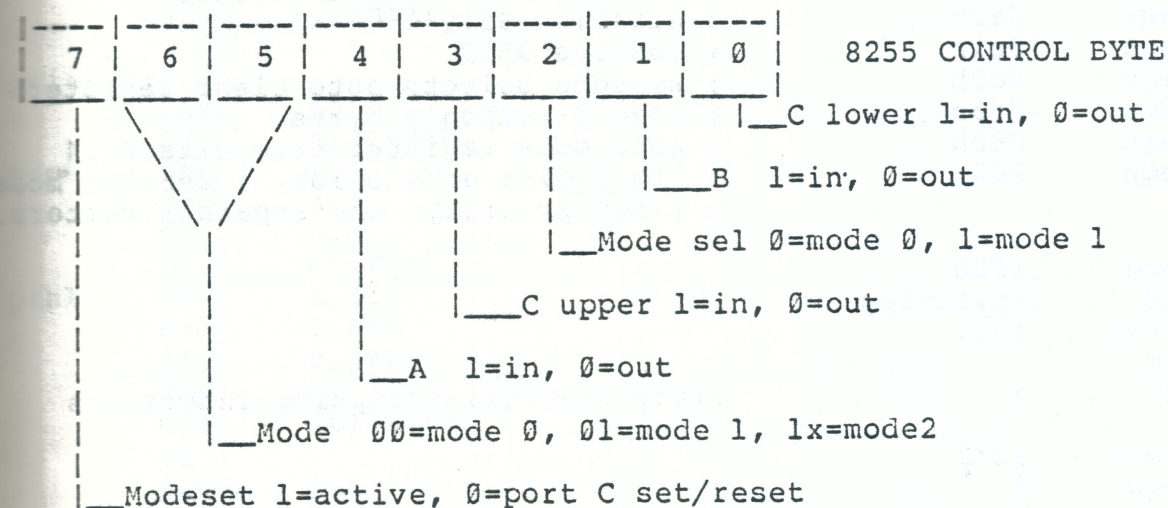
```
inch:   in      a,(1)          ;get status byte
        and     01             ;mask out char avail bit
        jr      z,inch         ;loop back til char there
        in      a,(0)          ;read character
        and     7fh            ;strip off parity bit (if any)
        ret                     ;return char in A
```

8255 PARALLEL I/O

The versatile Intel 8255 chip is used for the parallel I/O ports. This device can be configured in a bewildering variety of modes through software control, but for most typical applications, more output than input is usually needed, and latched data is often necessary. Thus (unless you make a few wiring changes), this board is configured to have the 8255 operate in the latched mode with one input port and two output ports. The outputs are buffered through 74LS244 drivers, which can sink 24 milliamperes, thus most loads can be driven directly.

The following table gives the standard setup for the 8255. See the 8255 data sheets for more information.

IO PORT	8255 Channel	Direction	Connector
08	A	OUTPUT	J5
09	B	OUTPUT	J5
0A	C	INPUT	J5
0B	-	MODE CONTROL	--



Example: To configure the 8255 for channel C input, and channels A and B output,

```
ld    a,89h          ;mode 0, A & B out, C in
out   (0bh),a
```

From here on, just do normal IN and OUT operations on ports 08 and 09 (output), and port 0A (input).

## 9519 Interrupt Controller

Here again, one must study the manufacturer's data sheet for a full understanding of this marvelously capable but complex LSI device.

A program for a typical setup for interrupts is given below to aid in understanding and serve as a guide.

; Test Program to type out level of interrupt received when bus pin  
; is grounded by a probe.

; 9519 interrupt controller port definitions:

ircdat equ 0EH ; data port  
irccmd equ ircdat+1 ; command port  
ircsts equ irccmd ; status port

; 9519 command definitions:

mskreg equ 0b0h ; selects mask register  
irrclr equ 040h ; clears i'rupt request register  
ircrst equ 000h ; resets interrupt controller  
ircdis equ 0a2h ; disables 9519  
ircen equ 0alh ; enables 9519  
irrato equ 0c0h ; selects selects auto clear register  
ircrsp equ 0f0h ; load 3 response bytes  
setmod equ 080h ; sets mode register from bits 0..4  
ircmod equ 000h ; irq & gint active low, interrupt mode  
; fixed priority, and separate vectors.

start: org 100h  
ld sp,stktop  
call init  
ei  
jr \$ ;loop forever, accepting interrupts

stktop defs 32\*2  
equ \$

int0: ; interrupt level 0 handler

;   
ld a,'0'  
jp printit

int1: ; interrupt level 1 handler

;   
ld a,'1'  
jp printit

int2: ; interrupt level 2 handler

;   
ld a,'2'  
jp printit

```

int3:  ; interrupt level 3 handler
;
ld     a,'3'
jp     printit

int4:  ; interrupt level 4 handler
;
ld     a,'4'
jp     printit

int5:  ; interrupt level 5 handler
;
ld     a,'5'
jp     printit

int6:  ; interrupt level 6 handler
;
ld     a,'6'
jp     printit

int7:  ; interrupt level 7 handler
;
ld     a,'7'
jp     printit

printit:
push   af
ld     hl,1000h
prn0:  dec     hl
ld     a,l
or     a,h
jr     nz,prn0
prn1:  in      a,(l)
and    a,04h
jr     z,prn1
pop    af
out    (0),a
ei
ret

init:  ; initialize system upon cold start
;
di
im0
ld     hl,initbl      ; HL := ^I/O initialization tables
call   puttbl         ; initialize everything
ret     ; exit (system will enable i'rups)

puttbl: ; output I/O initialization tables 'til zero length table
; tables are assumed to have the following format:
;
;      defb    length  ; length of data field
;      defb    port    ; port to send data to
;      defb    data1,data2,...,datan  ; data field
;

```

en  
de  
of  
ma  
co  
ta

to  
sy  
th

te  
ne  
se

Al  
out

tbl

```

; Upon entry:  HL := ^ first table to output
;
ld      a,(hl)          ; A := length
or      a
ret     z               ; done if length=0
ld      b,a
inc     hl
ld      c,(hl)          ; C := port adr, B := length
inc     hl
otir    ; output table data
jr      puttbl

initbl  equ      $
; interrupt controller initialization tables

defb    1,irccmd,ircrst      ; issue reset command

defb    1,irccmd,ircrsp or 0 ; select level 0 response
defb    3,ircdat            ; - interrupt handler 0
call    int0

defb    1,irccmd,ircrsp or 1 ; select level 1 response
defb    3,ircdat            ; - interrupt handler 1
call    int1

defb    1,irccmd,ircrsp or 2 ; select level 2 response
defb    3,ircdat            ; - interrupt handler 2
call    int2

defb    1,irccmd,ircrsp or 3 ; select level 3 response
defb    3,ircdat            ; - interrupt handler 3
call    int3

defb    1,irccmd,ircrsp or 4 ; select level 4 response
defb    3,ircdat            ; - interrupt handler 4
call    int4

defb    1,irccmd,ircrsp or 5 ; select level 5 response
defb    3,ircdat            ; - interrupt handler 5
call    int5

defb    1,irccmd,ircrsp or 6 ; select level 6 response
defb    3,ircdat            ; - interrupt handler 6
call    int6

defb    1,irccmd,ircrsp or 7 ; select level 7 response
defb    3,ircdat            ; - interrupt handler 7
call    int7

defb    1,irccmd,setmod
defb    1,ircdat,ircmod
defb    1,irccmd,irrato
defb    1,ircdat,0ffh      ; and set for all auto clear

```

```

defb 1,irccmd,mskreg      ; select mask register
defb 1,ircdat,00000000b  ; and enable all levels

defb 1,irccmd,ircen      ; arm i'rupt controller
defb 0                   ; end of initialization tables
end

```

en  
de  
of  
ma  
co  
ta

to  
sy  
th

te  
ne  
se

Alt  
out

tbl

## THEORY OF OPERATION

### Bus Signals:

The ZOBEX CPU board uses the following IEEE S-100 bus signals:

A0 - A18	High-true address lines
sOUT (45)	High-true, indicates that the current cycle is an output cycle, and that the address bus has the address of an output device.
sINP (46)	High-true, indicates that the current cycle is an input cycle, and that the address bus has the address of an input device.
sINTA (96)	High-true, indicates that the current cycle is an interrupt acknowledge cycle, and that the data bus is in the input mode.
sWO* (97)	Low-true, indicates that the current cycle is a transfer of data from the current bus master to a slave (memory or I/O).
sM1 (44)	High-true, indicates that the present bus cycle is for instruction read-up (followed by refresh).
sMEMR (47)	High-true, indicates that the data bus will be used for memory read.
sHLTA (48)	Results from execution of a halt instruction.
pWR* (77)	Low-true write signal, indicates that the data bus has stable data available.
pDBIN (78)	High-true, indicates the data bus is expecting data from the currently addressed slave.
pHLDA (26)	High-true, indicates that the data and address buses are in the high-impedance state. Appears in response to a HOLD signal (perhaps from a DMA device).
pSYNC (76)	High-true, indicates the start of a bus cycle.
RDY (72)	High-true, lets processor run as long as this line is high.

RDY (3) Same definition as RDY.

ROC\* (99) Low-true. Asserted from the CPU to indicate a master reset condition.

RESET\* (75) Low-true, indicates a total system reset has been commanded, usually from a front-panel switch closure to ground.

D0 - D7 High-true data in/out bus

VI0\* - VI7\* Low-true vectored interrupt lines.

NMI\* (12) Low-true non-maskable interrupt.

INT\* (73) Low-true external interrupt signal to CPU.

HOLD\* (74) Low-true, requests CPU to release bus.

DODSB\* (23) Low-true tri-states the data bus.

ADSB\* (22) Low-true, tri-states the address bus.

CDSB\* (19) Low-true, tri-states the control bus.

SDSB\* (18) Low-true, tri-states the status bus.

CLOCK (49) 2 MHz timing signal.

Phi (24) System clock (master timing signal).

REQ\* (65) Z80 memory request, for compatibility with older boards which need it.

RFSH\* (66) Z80 refresh signal, for compatibility with older boards which need it.

WRT (68) Memory write signal.

en  
de  
of  
ma  
co  
ta

to  
sy  
th

te  
ne  
se

All  
out

tbl

## CIRCUIT DETAILS

Refer to page 1 of the schematic diagrams.

### System Clock

The 16 MHz crystal oscillator at IC7 feeds a 74LS161 counter, which in turn produces a 4 MHz output on pin 13 and a 2 MHz output on pin 12. The state of the 2/4MHz switch at IC 36 is clocked into flip flop IC24 by the 2 MHz signal. If the switch is open, meaning 4 MHz, then the Q output of IC24 will be high, signifying 4 MHz clock. The complement output on pin 6 will be low. Thus in this case, pin 9 of AND gate IC39 will be enabled and the 4 MHz signal appearing on pin 10 will be gated through. In a similar manner, if the switch is closed, then the lower AND gate at IC39 will be enabled, and the 2 MHz clock will be gated through. The output of IC39 pin 6 is thus either 2 or 4 MHz, and is applied to the S-100 bus pin 24 through IC20. This signal is also used on the board for various timing purposes, and appears on pin 6 of the Z80.

### System Reset

When bus pin 75 is grounded by a front panel switch, the switch closure is de-bounced by the resistor-diode-capacitor network, inverted to a high at IC37 pin 2, and used to reset the latch at IC35. The reset signal is also distributed in true and inverted form to other places on the board.

### Power-on Jump Circuit

After a system reset, the Z80 will begin reading instructions from address 0000. The RESET\* signal applied to pin 13 of IC28 will set this power-on jump flip flop, forcing pin 9 low, and enabling the tri-state driver at IC34. Since the inputs to this driver are all low, this will force lows on the bi-directional data bus, which goes to the Z80 data bus pins. Thus the Z80 will be forced to read a 00 instruction byte, which is a NO-OP. The Z80 will then increment the address lines and continue to read-up and execute 00 bytes until address bits 12, 13, 14, and 15 match the setting of the SW36 power-on jump address switch. At this time the 7485 comparator at IC23 will output a high on pin 6, indicating a true comparison, which will complete the enabling of the AND gate at IC17 pin 8, which will drive pin 10 of IC28 low, and reset the power-on flip flop. This will remove the enable from IC34, and allow the Z80 to read an instruction from the selected address.

### EPROM Enable

In addition to clearing the power-on flip flop, IC17 pin 8 also controls the activation of the EPROM at IC12. A high on IC17 pin 11 indicates a memory read operation, and a high on pin 10 indicates that the software has NOT disabled the EPROM by clocking a high onto pin 3 of IC35 (this is done by outputting any byte to port 0C). Thus any memory read operation occurring within the 4K address space selected by the switch at IC36 will activate the EPROM by

placing a low on pin 20. This causes data or instructions from the EPROM to be output to the bi-directional data bus and read by the ISS.

### Data Bus Drivers and Receivers

The 74LS244 chips at IC46 and 47 provide the interface between the S-100 data-in and data-out buses and the bi-directional ISS data bus on the board. The RD\* signal from the Z80, buffered through IC22 pin 3, controls the enabling of these driver chips, unless prevented by the data bus disable signal generated at IC31 pin 6 on page 2.

### Address Bus Drivers

The 74LS244 drivers at IC43 and 44 drive the 16 bits of address from the Z80 onto the S-100 bus, unless disabled by the presence of a low on bus pin 22, the address disable line.

### pSYNC.

All S-100 bus cycles begin with a pSYNC signal. Accurate generation of pSYNC and pSTVAL\* signals is an important part of conformity with the IEEE bus standard. pSYNC is driven by IC 42 pin 14, which is in turn driven by flip flop IC28 pin 5. The flip flop is set through IC30 pin 3 by a signal derived from Z-80 signals MRQ\*, IORQ\* and RFSH\* such that a bus cycle will be started by either memory request or I/O operations from the Z-80, provided the RFSH\* signal is not active. A negative edge pulse generator made up of delay IC51 and AND gate IC18 will generate a narrow pulse when the phase terminates. The pSYNC flip flop is cleared and the pSYNC signal thus terminated by this end of phase pulse on pin 6 of IC18. This circuit meets all the IEEE requirements at all clock speeds while avoiding one-shots.

### CONTROL BUS.

The control signals, grouped together below the pSYNC pin, are generated directly from Z-80 chip outputs. Refer to the IEEE standard for the formal definition of these signals. Note that WAIT on pin 27 is provided, even though it is no longer specified. However, it is used by many popular memory and disk controller boards, so is included for this reason.

### STATUS BUS.

The S-100 status signals may be tri-stated when bus signal SDSB\* from pin 18 is low. The status signals indicate the type of bus cycle in process, and are placed on the data bus during pSYNC time through IC 45 in order to provide compatibility with older pre-standard boards such as the Cromemco Tuart.

sMEMR, indicating a memory read cycle, is generated at IC29 pin 4 from the memory request and read signals from the Z-80. The sOUT signal, indicating that the cycle in progress is an output

operation, is generated at IC29 pin 13 from the Z-80 IORQ\* and WR\* signals. Similarly, the SINP signal is generated at IC13 pin 1 from the Z-80 IORQ\* and RD\* signals. The SINTA signal, indicating that the current cycle is an interrupt acknowledge operation, is generated at IC16 pin 8 from a combination of IORQ\* and M1\* signals from the Z-80 (an otherwise illegal combination). The SWO\* bus signal, intended to be an early indication of a write operation, is generated at IC30 pin 6 by the Z-80 WR\* signal or any memory request operation which is not a read or a refresh.

Memory write strobe MWRT, which must be generated at only one place in the system, is produced at IC29 pin 10 by the condition WR\* and NOT SOUT.

Now please refer to page 2 of the schematic diagram.

### Vectored Interrupt Controller

The use of the AMD 9519 interrupt controller chip gives the ZOBEX CPU an unusual degree of power and flexibility in this area. A power-on sequence disables the 9519 by automatically writing zeroes into all the internal registers except the mask register, which is set to ones. Thus no interrupt requests will be recognized until the software initializes the 9519. The initializing is done by outputting data and control bytes to ports 0E and 0F, respectively. Chip select on pin 1, together with the A0 bit on pin 27 determine the port selection. After being properly conditioned by the software, the 9519 IEO, INT\*, PAUSE\*, and RIP\* lines will all be high.

When an interrupt occurs, through momentary grounding of one of the 8 vectored interrupt lines on the S-100 bus, the INT signal on pin 17 will be applied through pin 12 of OR gate IC30 and supplied to pin 16 of the Z80. This causes the Z80, after finishing the current instruction, to enter an interrupt acknowledge cycle, which is indicated by the assertion of both M1\* and IORQ\*. These signals are combined in the AND gate at IC16 so that pin 8 going low indicates the presence of an interrupt acknowledge bus cycle. Interrupt acknowledge pulses to the 9519 are supplied on pin 26 from pin 6 of OR gate IC52. The response in process signal (RIP\*) is applied through the diode or gate to pin 4 of IC52.

The first IACK\* pulse received by the 9519 causes selection of the highest priority unmasked interrupt which is pending, and generates the RIP\* output signal. Depending on the programmed state of the 9519, it will accept 1, 2, 3, or 4 IACK\* pulses, one for each response byte transferred to the CPU. The PAUSE\* output goes low when the first IACK\* is received, and remains low until RIP\* goes low. PAUSE\* is used to momentarily place the CPU in a wait state to stretch the acknowledge cycle and to allow the control timing to automatically adjust priority resolution delays in the interrupt system.

The second, third and fourth response bytes do not cause PAUSE\* to go low. The interrupt response bytes, to be read by the CPU, are taken

from the 9519's internal memory and placed on the bi-directional data bus during the INTA\* time. The interrupt cycle is completed as soon as the last byte has been transferred to the CPU.

### Baud Rate Generation

Timing signals for the DARTs and the CTC are generated by the crystal oscillator at IC7 at a frequency of 9.8304 MHz. This is fed to the 74LS161 down counter at IC25, generating frequencies of 1.2288 MHz on pin 12, and .6144 MHz on pin 11. The CTC is addressed to I/O ports 04, 05, 06, and 07 for channels 0, 1, 2, and 3, respectively. I/O operations on ports 04 through 07 are decoded by IC32 pin 14, and fed to the chip select input of the CTC on pin 16. A0 and A1 signals on pins 18 and 19 complete the address decoding of these ports. The CTC must be properly conditioned by the software, subsequent to a system reset, in order that the outputs on pins 7, 8, and 9 are at the desired baud rate.

For example, if it is desired to have a 9600 baud rate for serial channel A and the software has configured the DART at IC11 to require a clock of 32 times the baud rate, then pin 7 of the CTC must output a frequency of .3072 MHz. This can be obtained by programming the CTC to divide the input frequency of 1.2288 MHz by 4. In a similar manner, serial channel B receives its clock from CTC pin 8, and serial channels C and D receive their clocks from CTC pin 9. Channel 3 of the CTC is used for periodic clock interrupts to the CPU. Since the CTC prescaler can be programmed to divide by as much as 256, and the timer channel can divide by another 256, periodic interrupts as low as approximately 10 Hz may be generated.

### Daisy-chain Interrupts

The CTC, the two DARTs, and the 9519 are all in a daisy-chained interrupt priority network, with the priorities arranged in the same order. Any device in the chain that has an interrupt pending for service forces its IEO output to go low. Recall that the interrupt acknowledge cycle from the Z80 is indicated by both ICRQ\* and M1\* being low. Interrupt devices are prevented from changing state while M1\* is low, and when IORQ\* is low the highest interrupt priority requester will place its interrupt vector on the data bus and set its internal interrupt under service latch. The software may then determine from whence the interrupt came.

### Serial I/O Channels

Serial channels A and B are contained in the DART at IC11. This chip occupies I/O ports 0, 1, 2, and 3, and may be set up in a wide variety of modes by the software. Inputs and outputs from channel A are applied to J3 at RS-232 levels for direct connection to a terminal, while section B on J4 includes buffered RS-232 mode control signals.

The DART at IC10, which contains serial ports C and D, is essentially the same, except that ports 10, 11, 12, and 13 are used, and both sections run at the same baud rate.

### Parallel I/O

The 8255 chip at IC14 is configured to provide 3 8-bit parallel channels. Four I/O ports are necessary for this device, 08, 09, 0A, and 0B, which correspond to channel A, channel B, Channel C in, and control register, respectively.

### Info-switch

The 8 position dipswitch at IC50 may be read through port 0C, to provide a method for the software to determine system configuration options which can be set with this switch. The 74LS244 line driver at IC38 provides the interface between the switch and the bi-directional data bus.

## The ZOBEX Monitor

The on-board EPROM has a number of features useful for booting and debugging programs at the hardware level. The commands are summarized below. For additional details, refer to the source code on the distribution disk.

```
#####;
ZOBEX monitor;
=====;
B: boot disk system;
=====;
This routine allows the user to boot a disk system;
using the zobex disk controller. The routine;
loads sector one, track zero into memory at loca-;
tion 0000h and, if successful, branches to location;
0000h to execute the program.;
syntax:;
  B[cr];
#####;
```

```
#####;
D: display memory in hex;
=====;
This routine displays the contents of memory in hex;
with the starting location on each line (within the;
specified range). 16 bytes per line max.;
syntax:;
  D<start address>,<end address>;
#####;
```

```

;#####;
;#                                           #;
;#           G: goto <adr>, optionally set breakpoints      #;
;#                                           #;
;#=====;
;#                                           #;
;# This command allows execution of another program          #;
;# while retaining some monitor control by setting           #;
;# breakpoints. To simply execute, type 'G<adr>[cr]'.        #;
;# To set a breakpoint trap, add the address(es) to          #;
;# the command; ie: G<adr>,<bkpt>[cr]. Two breakpoints      #;
;# are allowed, enough to satisfy most requirements.         #;
;# Once a breakpoint has been reached, the registers         #;
;# may be examined or modified. The program can then         #;
;# be continued by typing only a 'G[cr]' or another          #;
;# breakpoint could be implemented at that time by           #;
;# typing 'G,<bkpt>[cr]'.                                     #;
;#                                                           #;
;# * note: This is software controlled, and the break-      #;
;# point must occur on an instruction byte.                  #;
;#                                                           #;
;#####;

```

```

;#####;
;#                                           #;
;#           M: move a block of memory                        #;
;#                                           #;
;#=====;
;#                                           #;
;# This command moves mass amounts of memory from <1>        #;
;# to <2> to the address starting at <3>. This routine       #;
;# should be used with some caution as it could              #;
;# smash memory if carelessly used.                           #;
;#                                                           #;
;#           M<1>,<2>,<3>[cr]                                #;
;#                                                           #;
;#####;

```

```
#####;
#;
I: input from a port #;
#;
===== #;
#;
This routine allows examination of any input port. #;
#;
#;
I<n>[cr] #;
display the port <n> #;
#;
#####;
```

```
#####;
#;
O: output to a port #;
#;
===== #;
#;
This routine allows the user to send any value to #;
any output port. #;
#;
O<n>,<v>[cr] #;
output to port <n>, the value <v> #;
#;
#####;
```

```
#####;
#;
S: display/alter memory #;
#;
===== #;
#;
This routine allows both inspection of and modifi- #;
cation of memory on a byte by byte basis. It takes #;
one address parameter, followed by a space. The #;
data at that location will be displayed. If it is #;
desired to change it, the value is then entered. A #;
following space will display the next byte. A car- #;
riage return[cr] will terminate the command. The #;
system adds a crlf at locations ending with either #;
xxx0 or xxx8. To aid in determining the present #;
address, it is printed after each crlf. #;
#;
#####;
```

en  
de  
of  
ma  
co  
ta  
  
to  
sy  
th

te  
ne  
se

Alt  
out

tbl

```

;#####;
;#;
;# X: examine/modify cpu registers;#;
;#;#;
;#=====;#;
;#;#;
;# This routine allows displaying the user's cpu reg-;#;
;# isters. You may also use the register name;#;
;# after typing the "X".;#;
;# i.e. XA 00-;#;
;# The register may be skipped over, or modified, as;#;
;# with the "s" command.;#;
;#;#;
;# To display the normal system status, simply type;#;
;# "X[cr]". To display the additional z-80 registers,;#;
;# type "X'[cr]". To examine a single "prime" regis-;#;
;# ter, type the register identifier after the apos-;#;
;# trophe.;#;
;# i.e. X'X 0000-;#;
;#;#;
;# These register values are placed into the cpu upon;#;
;# executing any "Go" command [G].;#;
;#;#;
;#####;

```

### TROUBLE SHOOTING TIPS

The ZOBEX Z80 CPU board is supplied only in wired and tested form, and is fully warranted for six months. If you have problems after the warranty period has expired and wish to fix them yourself, the following thoughts should be borne in mind.

You have a substantial investment in expensive chips, so be CAUTIOUS and never remove or install a board in the bus until power has been off for at least ten seconds to let the large filter capacitors discharge.

Remove the CPU board.

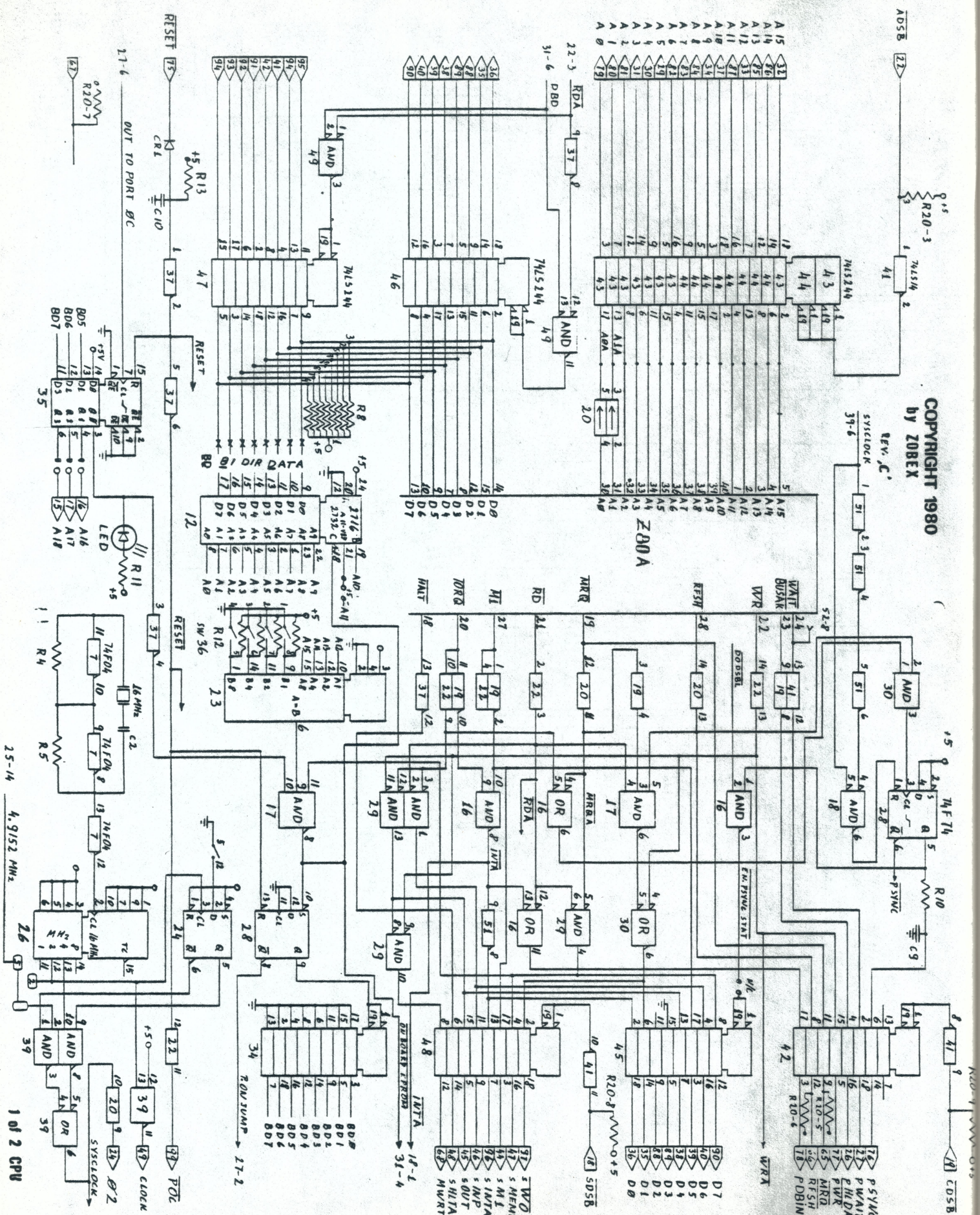
See that all the solder joints are clean, shiny, and smooth. Dull solder joints are cold solder joints and could cause problems. They should be resoldered. Make sure the solder does not overlap onto adjacent etches or pads.

Inspect the board for evidence of damage. Remove all other cards from the bus, and replace the CPU board. Apply power and check for normal voltages at the regulator outputs. If normal voltages are NOT observed, remove all chips from the board, plug it into the bus and apply power, watching carefully for smoke, smell, or heat. Then check for the proper voltages at the voltage regulator outputs (right-hand leads).

If all is well, replace the chips section by section until you find the one which is shorted. Replace the defective chip(s), re-install the other boards and try again. If you have trouble beyond this point, the usual trouble-shooting techniques apply to this board as well as any other.

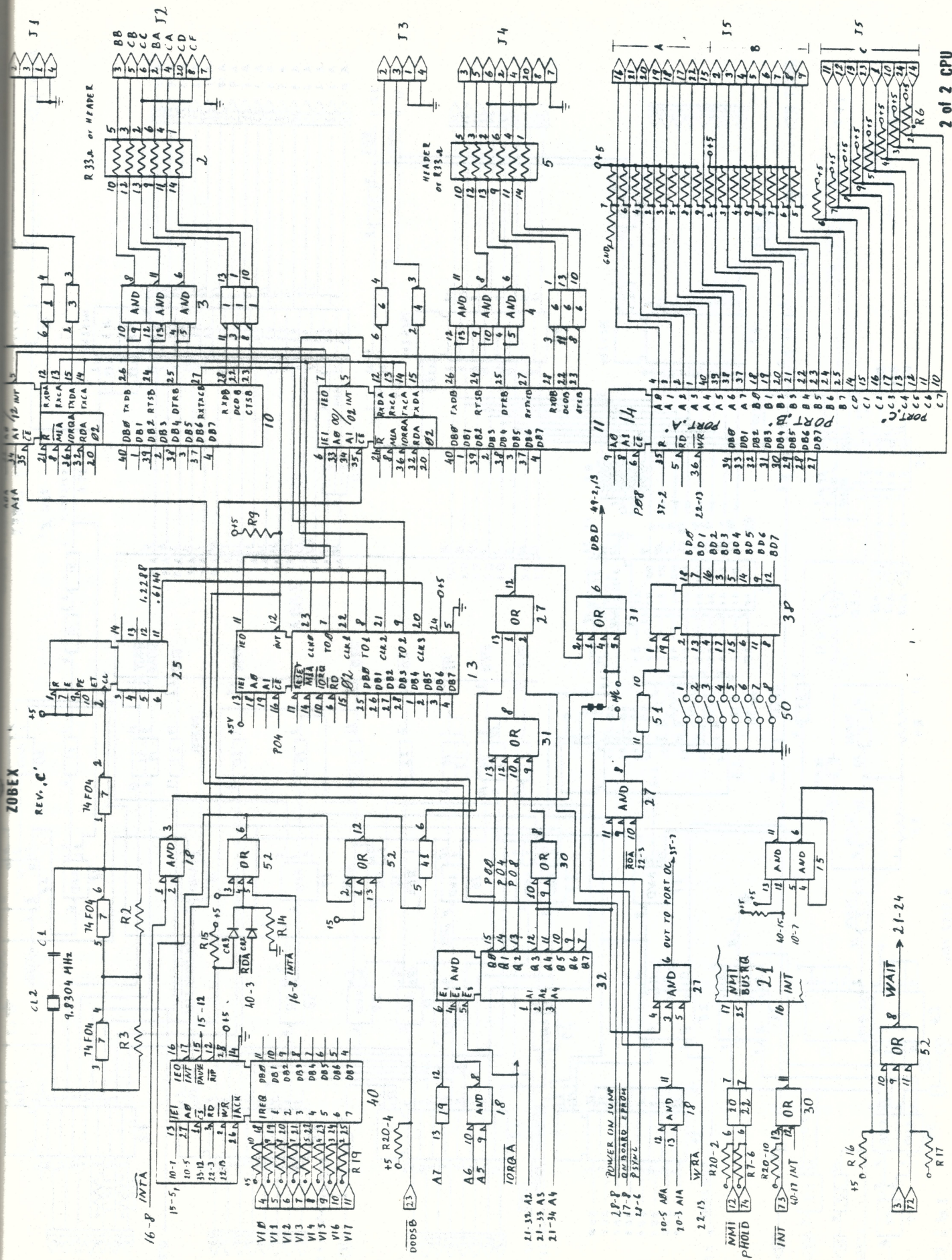
Test programs will be helpful, as will a wide-band oscilloscope. An effective Z-80 memory test is available from ZOBEX.

COPYRIGHT 1980  
by ZOBEX



# ZOBEX

REV. 'C'



2 of 2 CPU

# USR-100 MODEM

Data .....20

Command/Status..21

## PORT ASSIGNMENTS

FUNCTION	PORT	CONNECTOR
Serial Channel A		
Data in/out	00	J3
Status/control	01	
Serial Channel B		
Data in/out	02	J4
Status/control	03	
Serial Channel C		
Data in/out	10	J1
Status/control	11	
Serial Channel D		
Data In/out	12	J2
Status/control	13	
Parallel Channel A		
Data OUT	08	J5
Parallel Channel B		
Data OUT	09	J5
Parallel Channel C		
Data IN	0A	J5
Parallel Control A-C	0B	
Timer Channel 0	04 (Baud rate for serial channel A)	
Timer Channel 1	05 (Baud rate for serial channel B)	
Timer Channel 2	06 (Baud rate for serial channels C & D)	
Timer Channel 3	07 (Periodic clock interrupts)	
Info-switch		
Data IN	0C	
EPROM Disable		
OUT	0C	
Interrupt Controller		
Data in/out	0E	
Control	0F	

DD-FDC

Command/Status.....90

Track Register.....91

Seeker Register.....92

Data Register.....93